CSTDM09 - California Statewide Travel Demand Model

Model Development

Population

Final System Documentation: Technical Note

ULTRANS
Institute of Transportation Studies,
UC Davis
Davis, California

HBA Specto Incorporated
Calgary, Alberta

May 2011

Table of Contents

Tables:

Figures:

1. Introduction

This document provides background information and a description of the population data for the California Statewide Travel Demand Model (CSTDM09). The CSTDM uses the year 2000 as the base year for model estimation and the year 2008 as the model validation and calibration year. The person and household totals by transportation analysis zones (TAZ) are required as inputs to the CSTDM in both years. As with the 2000 population, we used a Population Synthesizer to derive a synthetic but realistic population using known target totals for several key characteristics, such as income class, dwelling type and age.

For the year 2000, we were often able to use data directly from the U.S. Census without much modification, because these data are available in small geographies for the required time period. For example, the population total, number of people making $0-$10,000/year, and number of people who own 2 cars can be found by block group for the year 2000. The block group totals can then be aggregated to the TAZ-level, and the population synthesizer can be run to cross-tabulate these totals.

In contrast, for the year 2008, these data have not been assembled as they were for the Decennial Census 2000. As a result, more processing of available data was necessary to acquire the appropriate targets and samples needed to run the Population Synthesizer.

The U.S. Census Public Use Microdata Sample (PUMS) 5% person and household data and Summary File 3 (SF3) data for the year 2008 are available from the website of the Census Bureau (http://factfinder.census.gov/home/en/acs_pums_2008_1yr.html). The PUMS data includes all persons in a household with both person and household attributes. However, PUMS data for California are only spatially located within 233 Public Use Microdata Areas (PUMAs). As a result, they cannot be used as the inputs for the CSTDM directly.

Census SF3 tables include all the attributes in PUMS and spatial location information, with resolution to the block group level. However, all the attributes are not cross-tabulated and, a results, they cannot be used as the inputs for the CSTDM directly either.

2. Population Synthesizer

The population synthesizer developed by John Abraham and Doug Hunt (HBA Specto) works by combining a trial population of households and altering it by switching new possible households in. If the match with the targets improves, the new household is kept. A detailed description of the algorithms used in this process is part of the detailed documentation of the population synthesizer (Appendix 1). The population synthesizer is capable of handling multiple nested geographies, of matching categorical totals or averages, and of weighting possible targets. The weighting capability is useful if some targets are considered to be more important than others, or if the scales differ (such as with an average income category).

In general, synthesizing the population consists of four steps: 1) creating sample tables or individual household records; 2) creating target tables or control totals for available geographies; 3) testing the goodness of fit; and 4) aggregating the synthesized population by traffic analysis zones (TAZ). To enhance the accuracy of the population synthesis, population is synthesized by PUMA. Each PUMA has a sample table and a target table.

2.1 Developing Samples

For the years 2000 and 2008 population syntheses, the PUMS data were used as the basis for samples. Figure 1 shows a snapshot of a sample table. These samples consist of housing units and persons. A composite sample record was created for each PUMS housing unit and the associated person record(s), with the totals for each of the targets. For instance, a housing unit of 2 people living in a 5 to 9 unit apartment building; a 38 year old factory worker and a 42 year old welder with annual income of $82,302 and 2

cars would become a record of 1 household of 2 persons, as well as 1 household in the $75 to 100K income category, 1 household living in a multifamily dwelling and 1 household with 2 cars, with 1 person 25-39 and 1 person 40-54 and 2 blue collar workers. (There would be 0 for every other value in the table in this case; there are 0 students, 0 households living in a single family dwelling unit, 0 1-person households and so on.)

	A	B	C	D	I	J	K	L	M	N	O	P
1	UniqueID	1_pers	2_pers	3_pers	type_sfd	type_att	type_mult	type_mh	inc_0_10	inc_10_25	inc_25_50	inc_50_75
2	3513665	1	0	0	0	0	1	0	0	1	0	0
3	3513665	1	0	0	0	0	1	0	0	1	0	0
4	3513665	1	0	0	0	0	1	0	0	1	0	0
5	3513665	1	0	0	0	0	1	0	0	1	0	0
6	3513665	1	0	0	0	0	1	0	0	1	0	0
7	3513665	1	0	0	0	0	1	0	0	1	0	0
8	3513665	1	0	0	0	0	1	0	0	1	0	0
9	3513665	1	0	0	0	0	1	0	0	1	0	0
10	3513665	1	0	0	0	0	1	0	0	1	0	0
11	3513665	1	0	0	0	0	1	0	0	1	0	0
12	3513665	1	0	0	0	0	1	0	0	1	0	0
13	3513665	1	0	0	0	0	1	0	0	1	0	0
14	3513665	1	0	0	0	0	1	0	0	1	0	0
15	3513665	1	0	0	0	0	1	0	0	1	0	0
16	3513665	1	0	0	0	0	1	0	0	1	0	0
17	3513665	1	0	0	0	0	1	0	0	1	0	0
18	3513665	1	0	0	0	0	1	0	0	1	0	0
19	3513665	1	0	0	0	0	1	0	0	1	0	0

Figure 1: Snapshot of a Sample Table

Because California is a large state, the population characteristics vary significantly from place to place; people from suburban Orange County may have very different characteristics than people from rural Humboldt County or downtown San Francisco. To reflect this, each PUMA was processed separately, with PUMS records for the PUMA in question (the “own” PUMA) and for nearby PUMAs. The weights provided for the PUMS records were respected; the “own” PUMA had the full household level weight applied, and the five nearest PUMAs (using a centroid-based straight line distance) had 0.2 of the housing unit level weight, rounded down with at least one record included. To do this, the records were simply duplicated; a housing unit with a weight of 14 would have 14 identical records in the sample file for the “own” PUMA, and 2 records in the sample files for the nearby PUMAs.

2.2 2000 Zonal Targets

For the 2000 population synthesis, all targets were available at the block group level, and thus, at the zone level. The targets were treated categorically (for example, rather using an average income, the number of households in seven income categories is represented). Because categorical totals are used, all weights were set at 1. The target totals used for the year 2000 are presented in Table 1.

Table 1: Year 2000 Target Totals

Type	Number of categories	Detail
Household size	7	1 person, 2 person, 3 person, 4 person, 5 person, 6 person, 7+ person
Dwelling type	5	Single family detached, single family attached, multifamily, mobile home, group quarters
Household income	7	0-10K, 10-25K, 25-50K, 50-75K, 75-100K, 100-150K, 150K+ (note: year 2000 dollars for both samples and targets)
Persons by age	10	0-4, 5-15, 16-18, 19-21, 22-24, 25-39, 40-54, 55-64, 65-74, 75+
Auto ownership	6	0 cars, 1 car, 2 cars, 3 cars, 4 cars, 5+ cars
Workers by occupation	6	Managerial/Professional, Business/Office, Sales/Food/Entertainment, Service/Health/Education, Blue Collar, Military
Students by school type	3	Kindergarten to grade 8, grade 9 to 12, post secondary
Number of Rooms	9	1 room, 2 rooms, 3 rooms, 4 rooms, 5 rooms, 6 rooms, 7 rooms, 8 rooms, 9 or more rooms
Number of Bedrooms	6	No bedrooms, 1 bedroom, 2 bedrooms, 3 bedrooms, 4 bedrooms, 5 or more bedrooms

These targets were all derived from SF3 totals, provided at the block group and aggregated to the zonal level as needed. See the employment model documentation for a detailed discussion of work occupation groups.

2.3 2008 Zonal Targets

The target totals in Table 1 were not available for the year 2008 by block group, or by any geography that could easily be aggregated to the TAZ level. Instead, we used a

combination of data sources to provide population totals at the TAZ-level, and the other socioeconomic targets at the PUMA-level. While it is preferable to have all of the 2008 zonal targets by TAZ, the available data and geography did not allow for the same table design as for the year 2000.

2.3.1 Creating control totals using ACS data

American Community Survey (ACS), Metropolitan Planning Organization (MPO), county and the Department of Finance (DOF) data sources were all used to generate the target tables. The following targets were available at the PUMA level:

1. Household size
2. Dwelling type
3. Household income
4. Persons by age
5. Auto ownership
6. Workers by occupation type
7. Students by school type.

The targets were all treated as categorical and all weights were set at 1. The number of categories and the description of targets are presented in Table 2.

Table 2: 2008 Targets by PUMA

Variables	Target	Source table (ACS)
1 Person 2 Person 3 Person 4 Person 5 Person 6 Person 7 Person	household size	B25009. TENURE BY HOUSEHOLD SIZE - Universe: OCCUPIED HOUSING UNITS (owner-occupied plus renter-occupied)
Single Family Detached Single Family Attached Multi-Family Mobile Home Group Quarters	dwelling type	B25032. TENURE BY UNITS IN STRUCTURE (owner-occupied plus renter-occupied); B26001. GROUP QUARTERS POPULATION
Income 0-10 Income 10-25 Income 25-50 Income 50-75 Income 75-100 Income 100-150 Income 150+	household income (1,000's)	B19001. HOUSEHOLD INCOME IN THE PAST 12 MONTHS (IN 2008 INFLATION-ADJUSTED DOLLARS)
Age 0-4	person by age	B01001. SEX BY AGE

Age 5-15 Age 16-18 Age 19-21 Age 22-24 Age 25-39 Age 40-54 Age 55-64 Age 65-74 Age 75+		(male plus female)
0 Cars 1 Car 2 Cars 3 Cars 4 Cars 5+ Cars	Auto-Ownership	B08201. HOUSEHOLD SIZE BY VEHICLES AVAILABLE
Managerial and Professional Business and Office Sales, Food, and Entertainment Service, Health and Education Blue Collar Military	Worker's Occupation	C24020. SEX BY OCCUPATION FOR THE FULL-TIME, YEAR-ROUND CIVILIAN EMPLOYED POPULATION 16 YEARS AND OVER (male plus female); C23001. SEX BY AGE BY EMPLOYMENT STATUS FOR THE POPULATION 16 YEARS AND OVER (for Armed Forces,
Kindergarten – Grade 8 High School (Grade 9 – Grade 12) Post-secondary Education	Students by Advancement Level	B14001. SCHOOL ENROLLMENT BY LEVEL OF SCHOOL FOR THE POPULATION 3 YEARS AND OVER

Table 3: Occupation Categories

2008 population synthesizer categories	ACS occupation categories
Managerial and Professional Occupations	Management occupations
	Community and social service occupations
	Legal occupations
	Computer and mathematical occupations
	Architecture and engineering occupations
	Life, physical, and social science occupations
Business and Office Occupations	Business and financial operations specialists
	Office and administrative support occupations
Service, Health and Education Occupations	Education, training, and library occupations
	Personal care and service occupations
	Healthcare practitioners and technicians occupations
	Healthcare support occupations
	Protective service occupations
Sales, Food and Entertainment Occupations	Food preparation and serving related occupations
	Sales and related occupations
	Arts, design, entertainment, sports, and media occupations
Blue Collar Occupations	Building and grounds cleaning and maintenance occupations
	Farming, fishing, and forestry occupations
	Construction and extraction occupations
	Installation, maintenance, and repair occupations

	Production occupations
	Transportation and material moving occupations
Military Workers	Military occupations, and all military workers of any occupation

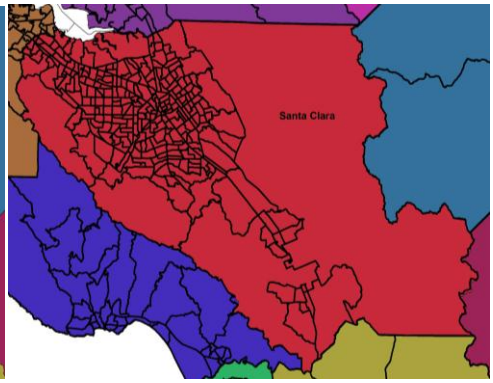
2.3.3 Creating Zonal Targets Using MPO and County Data

The socioeconomic targets described above are provided by PUMS for the PUMA geography, which for most of California is quite a bit larger than the TAZ (see Figure 2). For a more accurate depiction of the demographic characteristics of the population, we needed, at a minimum, the 2008 population total for each TAZ. As with the other 2008 socioeconomic data, population totals were unavailable for a geography small enough to aggregate to the TAZ-level. The U.S. Census publishes population data by city, and, along with the Department of Finance, by county. Fortunately, we could also access data at the more local level of MPO and county.

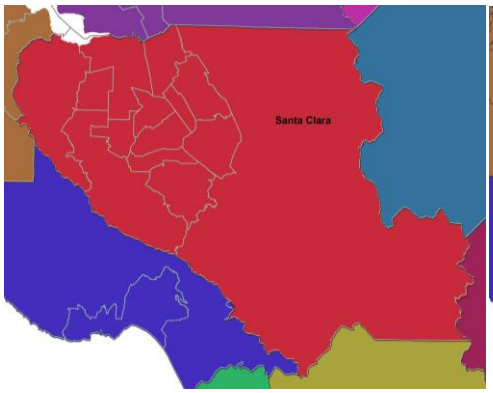
We chose to use Metropolitan Planning Organization (MPO) data if possible for several reasons: (1) local demographers were likely to be more attuned to differences in population growth for smaller geographies; (2) it was important to have MPO and county support of the population estimates used for the model; and (3) the MPO population data came in the format of smaller geographies, which allowed us to better estimate the difference in population for our geographies through aggregation rather than dissolution. We were able to attain TAZ GIS data and socioeconomic data for 39 of 58 counties. The socioeconomic data varied by MPO and county, but the only component to be added to the population synthesizer was the population total for the year 2008. If 2008 data were unavailable, the closest year available was used instead.

A. Santa Clara County

B. CSTM TAZ Boundaries



C. PUMA Boundaries



D. MTC TAZ Boundaries

E. Close up of CSTM and MTC TAZ Boundaries showing discrepancy

Figure 2: Geography Comparison: Santa Clara County

2.2.4 Joining MPO/County TAZ Boundaries and the CSTDM09 TAZ boundaries

Because the MPO/County TAZ boundaries and the CSTDM09 TAZ boundaries were not an exact match, some geoprocessing was necessary to join the two sets of data. For all MPO/County TAZ data, the geoprocessing technique was identical, a multi-step process as follows:

1. Prepare three data sets:
    i. **CSTM_TAZ** GIS shapefile
    ii. **MPO_County_TAZ** GIS shapefile
    iii. **2000_Census_Block_Group** GIS shapefile
    b. Each shapefile is reduced to the smallest common extent using the ArcGIS Geoprocessing tool “Clip”
    c. A new field is added to each shapefile, “Area (sq. mi)” and the area in square miles is calculated for all records using the “Calculate Geometry” tool
2. Find all **CSTM_TAZ** with completely-nesting **MPO_County_TAZ** (98% nesting is considered complete)
    a. Join **CSTM_TAZ** GIS and **MPO_County_TAZ** GIS using ArcGIS Geoprocessing tool “Union”. Output is new GIS shapefile **CSTM_MPO_Union** which contains all intersecting polygons of the two initial shapefiles
    b. Calculate the area of each new polygon in the **CSTM_MPO_Union** GIS shapefile
    c. Calculate the proportion of **CSTM_MPO_Union** GIS shapefile area to the **MPO_County_TAZ** GIS shapefile, but dividing the **CSTM_MPO_Union** area by the **MPO_County_TAZ** area
    d. Select only the polygons where the proportion is greater than or equal to 98%
    e. Export selected polygons to a new shapefile, **MPO_nest**
    f. Export the non-nesting polygons **CSTM_MPO_Union_no_nest**
    g. Aggregate the nesting polygons of **MPO_nest** to the CSTM TAZ-level using ArcGIS geoprocessing tool “Dissolve”

i. Sum the fields for the “Union Area (sq. mi)” and the “MPO/County Population” for each unique “CSTM TAZ ID”
ii. Name the new aggregated shapefile **MPO_nest_Dissolve**

h. Add a new field in **MPO_nest_Dissolve** that calculates the proportion of summed “Union Area (sq. mi)” to “CSTM Area (sq. mi)
i. Select only the polygons where the proportion is greater or equal to than 98%
j. Export selected polygons to a new shapefile, **CSTM_nest**

3. Select the **MPO_County_TAZ** that fit entirely within **CSTM_TAZ**, but only account for part of the total **CSTM_TAZ**
    a. Select polygons with a proportion of total “Union Area (sq. mi)” to “CSTM Area (sq. mi)” that is less than 98%
    b. Export selected polygons to a new shapefile, **MPO_nest_Incomplete**
4. For the remaining areas where **MPO_County_TAZ** are crossed by a **CSTM_TAZ** boundary, we used 2000 Census Block Group population to allocate how much MPO/County population should be assigned to the CSTM TAZ.
    a. Find the block group population totals by MPO/County TAZ using area; this essentially breaks apart all of the block group boundaries into small pieces using MPO/County TAZ boundaries, multiplies each fragment’s area by the population, and adds up all the fragments using the MPO/County TAZ boundaries
        i. Intersect the boundaries of **MPO_County_TAZ** and **2000_Census_Block_Group** using ArcGIS Geoprocessing tool “Union”. Output is new GIS shapefile **MPO_BG_Union**
        ii. Calculate the proportion of **MPO_BG_Union** GIS shapefile area to the **2000_Census_Block_Group** GIS shapefile, but dividing the **MPO_BG_Union** area by the **2000_Census_Block_Group** area and enter result for each record in a new field
        iii. Multiply the calculated proportion for each record by the 2000 Census Block Group population and enter results in a new field

iv. Aggregate the total 2000 Census Block Group population by **MPO_County_TAZ** using ArcGIS Geoprocessing tool “Dissolve” which results in a new shapefile with MPO/County TAZ ID and the sum of **2000_Census_Block_Group** population. Output is new GIS shapefile **MPO_BG_Union_Dissolve.**

b. Find the proportion of block group population to the total block group population of the MPO TAZ area
    i. Join the **MPO_BG_Union_Dissolve** table to the **MPO_BG_Union** table using the MPO/County TAZ ID as the primary key
    ii. Calculate the proportion of block group population to the sum of the block group population all of the disaggregated block groups that were re-aggregated by MPO/County TAZ in the previous step
c. Multiply the MPO/County TAZ Population by the block group population proportion
d. Next, aggregate the fragments of the MPO/County TAZ layer segmented by CSTM boundaries (now with MPO population values per fragment) to CSTM TAZ
    i. Select all of the fragments of **MPO_BG_Union** with identical locations as **CSTM_MPO_Union_no_nest** by using the “Select by Location” tool
    ii. Export selected polygons to a new shapefile: **MPO_BG_union_no_nest**
    iii. Join the **CSTM_MPO_Union_no_nes**t shapefile to the **MPO_BG_union_no_nest** shapefile using the ArcGIS Geoprocessing tool “Union”. This will geographically identify which **MPO_BG_union_no_nest** records belong to each CSTM TAZ ID. The output is a new shapefile **Union_no_nests**.
    iv. Aggregate the fragments to the CSTM TAZ level by using the ArcGIS Geoprocessing tool “Dissolve” so that each CSTM TAZ ID has the sum of all MPO/County TAZ population fragments within

each CSTM TAZ boundary. The output is a new shapefile U**nion_no_nests_Dissolve**

5. Assemble the three types of MPO/County and CSTM boundary scenarios: complete nesting, incomplete nesting, or segmented by CSTM boundary
    a. Join the tables of **CSTM_nes**t, **MPO_nest_Incomplete** and U**nion_no_nests_Dissolve** to the table of **CSTM_TAZ**
    b. For the records that have a value for **CSTM_nes**t, simply copy the value to the matching **CSTM_TAZ** record.
    c. For the remaining records, add the values of **MPO_nest_Incomplete** and U**nion_no_nests_Dissolve**
    d. If Group Quarters are not included in the MPO/County population data, the year 2000 Group Quarters data from the Census was added to the population as an approximate account of current Group Quarters population.

The counties and MPOs with CSTM TAZ targets developed using MPO/County demographic data: MTC (Alameda, Contra Costa, Marin, Napa, San Francisco, San Mateo, Santa Clara, Solano and Sonoma), MCOG/Wine Country IRP (Lake, Mendocino), AMBAG (Monterey, San Benito and Santa Cruz), SACOG (El Dorado, Placer, Sacramento, Sutter, Yolo and Yuba), SCAG (Imperial, Los Angeles, Orange, Riverside, San Bernardino and Ventura), SANDAG (San Diego), Butte, Fresno, Humboldt, Kern, Kings, Merced, San Joaquin, San Luis Obispo, Santa Barbara, Shasta, Stanislaus and Tulare.

2.2.5 Development of Zonal Targets Using DOF Data

There are 19 counties from which we did not receive demographic data generated by a county or MPO. For the CSTM TAZ included in these counties, a different method of producing 2008 population targets was used. In general, these counties are more rural and sparsely populated than the zones using MPO and county data and thus have a smaller amount of additional people to distribute. Also, TAZ in rural counties tend to be larger, so there are fewer TAZ to distribute the additional population amongst. The “E-5 Population and Housing for Cities, Counties and the State, 2001-2010, with 2000

Benchmark," *Department of Finance, 2010*, Sacramento, DOF, provided the 2000 and 2008 population totals by county, incorporated cities, and balance of county.

Using ArcGIS, a comparison of the CSTM TAZ shapefile and Census Places shapefile provided a link between incorporated city population and CSTM TAZ. For cities that were spread over multiple TAZ and for the balance of county population, satellite imagery was used to aid in assigning population growth to TAZ. The CSTM TAZ with zonal targets developed using DOF data include: Alpine, Amador, Calaveras, Del Norte, Colusa, Glen, Inyo, Lassen, Madera, Mariposa, Modoc, Mono, Nevada, Plumas, Sierra, Siskiyou, Trinity, Tehama and Tuolumne.

3. Data Processing

These sample and target tables are prepared for the 233 PUMAs in California through an automated Python script. These tables are the inputs of the population synthesizer which is programmed in Java. The synthesizer generates a table which gives all the households (which are represented by the unique household serial number in PUMAs household table) assigned to a TAZ.

The population synthesis Java program was run for the 233 PUMAs in order; because this was being run once rather than "in line" with a model run, and runtime was unimportant, the MaxIterations was set to 10 million for each PUMA. The simulated annealing options were not used for the run. The resulting output samples were then loaded into the same database that held the SF3 and the PUMA data used to create the targets and samples; the table was used to link the raw PUMS records and the synthetic population. Thus, each TAZ in California has a fully synthesized set of households and people consistent with all the personal and household attributes known for that TAZ.

Appendix 1. Synthesizer Documentation

Documentation for SYNTHESIZER Program

John E. Abraham and J.D. Hunt, February 2006

Draft version 0.4

Introduction

This note describes the setup and use of the SYNTHESIZER software for generating synthetic populations.

Method

The software works to identify a list of units whose aggregate attribute values match a pre-specified set of corresponding target values. This list forms a synthetic population of such units consistent with the target values. Each unit included in this list is drawn from a sample of such units, with the potential that any particular unit in the sample is included in the list 0, 1 or more times as appropriate.

The software proceeds by iteratively considering adding a unit from the sample to the list, subtracting a unit from the list, or 'swaps' where a unit in the list is swapped out and a unit from the sample is swapped in. The match of the list to the target values is scored using a goodness-of-fit function.

The list is divided into subgroups, which are commonly used to specify geographic areas known as "zones" which are to contain portions of a population. The process works through the list subgroup by subgroup. For each subgroup first one of the three operations is selected with equal 1/3 probability (add, subtract or swap). In the case of subtract or swap a unit in the subgroup is randomly selected. In the case of add or swap a unit in the sample is randomly selected. The operation is then performed, and the magnitude of the improvement in the goodness-of-fit score is calculated. If the

goodness of fit improves the operation is kept. If the goodness of fit gets worse there is a less-than-1.0 probability that the operation will be kept, otherwise the operation will be undone.

It is possible to have the program start with a previously generated list. If no such previously generated list is available, an initial list is generated by randomly selecting enough units at random to reach or exceed target values for just one of the attributes for each subgroup.

The decision to keep an operation at any point includes a probabilistic component. Operations that lead to a worse goodness of fit will be more likely to be accepted early in the process than later in the process. This is what is termed a 'simulated annealing' algorithm – based on the idea that the program should be getting closer to the best possible match as the number of iterations increases and thus a non-improving swap is more likely to be detrimental rather than advantageous in the search for this best possible match.

The general formula used in the measurement of goodness-of-fit is:

$$gof = Sqrt(\Sigma_a \, weight_a^2 \cdot (list_a - target_a)^2)$$

where:

a = index of attributes whose aggregate values for the synthetic population list are to match a pre-specified set of corresponding target values
gof = goodness-of-fit, with values closer to 0 indicating a better fit (such that it might be appropriate to consider it a 'lack-of-fit' measure).
$weight_a$ = weight associated with attribute a
$list_a$ = aggregate value for attribute a for the list
$target_a$ = target value for attribute a

The weight associated with an attribute indicates the relative importance to be placed on achieving a match with regard to that attribute.

The formula used to assign the probability of accepting an operation that leads to a worse goodness of fit is:

$P = \exp(-\text{iteration}/\alpha)^{(\Delta\text{gof}^{\gamma})}$

where:

P = probability of accepting the operation
Δgof = change in goodness-of-fit associated with the operation
γ = parameter controlling influence of the size of the change in goodness-of-fit on the probability of making the swap, specified as *gofDifExponent* in the properties file for the program
iteration = number of operations that have been evaluated so far in the process
α = parameter controlling influence of the number of iterations on the probability of making the swap, specified as *coolingParameter* in the properties file for the program

It is common to set γ to 0.0 when first setting up the synthesizer, so that the probability of accepting an operation that leads to a worse goodness of fit is simply

$P = \exp(-\text{iteration}/\alpha)$

The number of iterations to be performed by the program is specified by the user as part of the inputs.

The target values for the attributes can be specified for individual subgroups of the population or for combinations of the subgroups. The program proceeds subgroup-by-

subgroup in its processing, with the number of iterations within each subgroup for each pass through the entire list of subgroup determined according to the comparative goodness-of-fit for the subgroup.

The formula used to establish the number of iterations for a given subgroup is:

$n_z = \iota \cdot gof_z + 1$

where:

z = index of subgroup
gof_z = goodness-of-fit for subgroup z
ι = parameter controlling influence of the goodness-of-fit value for a subgroup on the number of iterations for the subgroup as part of the current pass through the subgroups, called *iterationsPerZonalLackOfFit* in the properties file.

The program performs iterations on each subgroup, checking the total number of iterations every time a subgroup is processed, and terminating if the total number of iterations meets or exceeds the specified number of iterations. Upon termination the program reports the overall goodness-of-fit, the goodness of fit for each subgroup and for each target in each subgroup, and the resulting list of units comprising the synthetic population.

Inputs

The inputs to the program are provided in a set of computer files as follows:

1. A file referred to as the samplesTable (often called *Samples.csv*; the file must have the filename extension ".csv" but the name of the file is specified in the properties file, described below) contains a list of the full set of sample units. Each line contains all the attribute values for one unit.

A screenshot showing the format of this file is included below (Figure 3, Figure 4)). The first line contains a set of column headers indicating the attribute label for the values included on each remaining line of the file; these labels must match the corresponding labels for the target values specified in the targetsTable file as described below. (If there is a column label used here that does not match any column label in targetsTable, then the values are retained in the synthesized list of units but are not considered in any target matching, which allows the user to use such values in further calculations and to retain such values for subsequent application.)

The first column in the samplesTable must contain a number which uniquely identifies the sample. (Numbering the rows sequentially using the first column would suffice if no prior numbering system exists.)

The extension ".csv" refers to "comma separated value" and implies a text file that can be interpreted as a table. The rows in the table are separate lines in the text file, and the columns in the table are separated by commas. This type of file can be directly edited by a large number of programs, including database programs and spreadsheet programs such as Microsoft Excel.

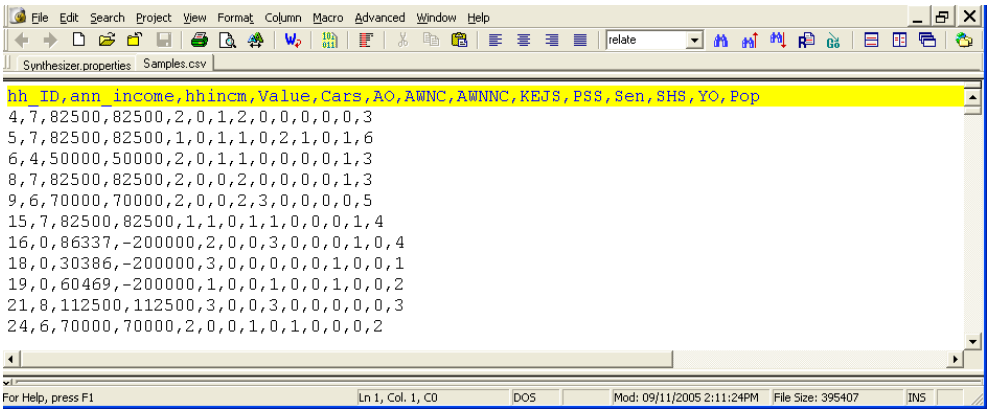
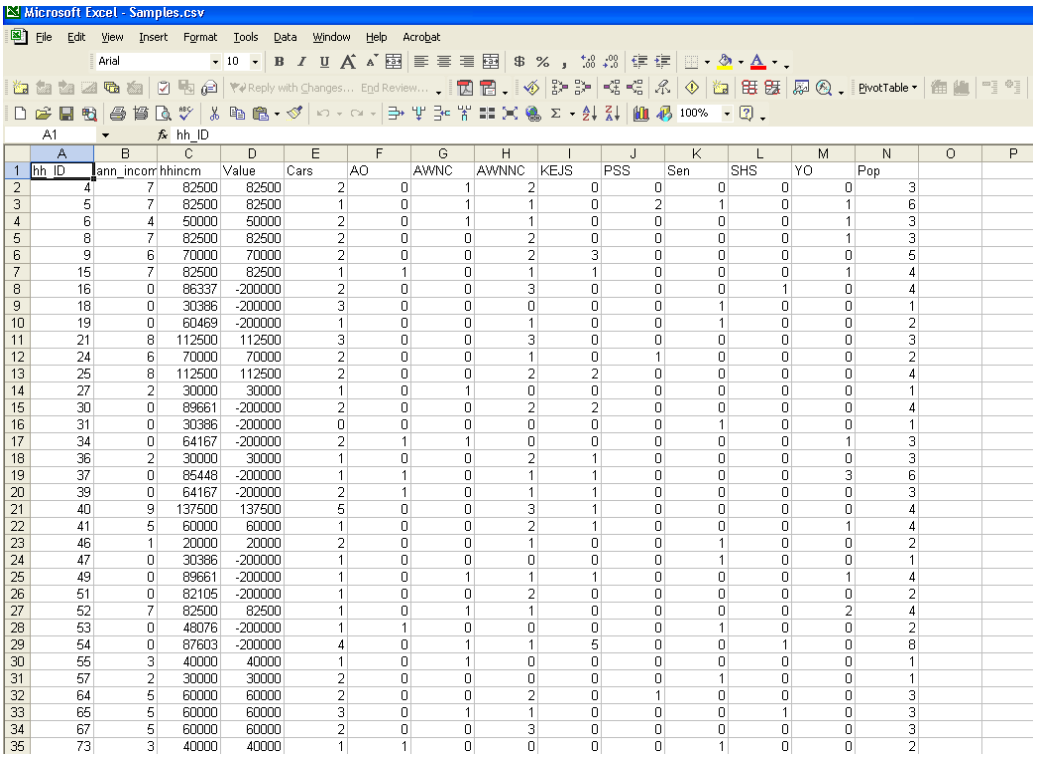


Figure 3: Editing the Samples.csv file in the text editing program UltraEdit.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	hh_ID	ann_incom	hhincm	Value	Cars	AO	AWNC	AWNNC	KEJS	PSS	Sen	SHS	YO	Pop		
2	4	7	82500	82500	2	0	1	2	0	0	0	0	0	3		
3	5	7	82500	82500	1	0	1	1	0	2	1	0	1	6		
4	6	4	50000	50000	2	0	1	1	0	0	0	0	1	3		
5	8	7	82500	82500	2	0	0	2	0	0	0	0	1	3		
6	9	6	70000	70000	2	0	0	2	3	0	0	0	0	5		
7	15	7	82500	82500	1	1	0	1	1	0	0	0	1	4		
8	16	0	86337	-200000	2	0	0	3	0	0	0	1	0	4		
9	18	0	30386	-200000	3	0	0	0	0	0	1	0	0	1		
10	19	0	60469	-200000	1	0	0	1	0	0	1	0	0	2		
11	21	8	112500	112500	3	0	0	3	0	0	0	0	0	3		
12	24	6	70000	70000	2	0	0	1	0	1	0	0	0	2		
13	25	8	112500	112500	2	0	0	2	2	0	0	0	0	4		
14	27	2	30000	30000	1	0	1	0	0	0	0	0	0	1		
15	30	0	89661	-200000	2	0	0	2	2	0	0	0	0	4		
16	31	0	30386	-200000	0	0	0	0	0	0	1	0	0	1		
17	34	0	64167	-200000	2	1	1	0	0	0	0	0	1	3		
18	36	2	30000	30000	1	0	0	2	1	0	0	0	0	3		
19	37	0	85448	-200000	1	1	0	1	1	0	0	0	3	6		
20	39	0	64167	-200000	2	1	0	1	1	0	0	0	0	3		
21	40	9	137500	137500	5	0	0	3	1	0	0	0	0	4		
22	41	5	60000	60000	1	0	0	2	1	0	0	0	1	4		
23	46	1	20000	20000	2	0	0	1	0	0	1	0	0	2		
24	47	0	30386	-200000	1	0	0	0	0	0	1	0	0	1		
25	49	0	89661	-200000	1	0	1	1	1	0	0	0	1	4		
26	51	0	82105	-200000	1	0	0	2	0	0	0	0	0	2		
27	52	7	82500	82500	1	0	1	1	0	0	0	0	2	4		
28	53	0	48076	-200000	1	1	0	0	0	0	1	0	0	2		
29	54	0	87603	-200000	4	0	1	1	5	0	0	1	0	8		
30	55	3	40000	40000	1	0	1	0	0	0	0	0	0	1		
31	57	2	30000	30000	2	0	0	0	0	0	1	0	0	1		
32	64	5	60000	60000	2	0	0	2	0	1	0	0	0	3		
33	65	5	60000	60000	3	0	1	1	0	0	0	1	0	3		
34	67	5	60000	60000	2	0	0	3	0	0	0	0	0	3		
35	73	3	40000	40000	1	1	0	0	0	0	1	0	0	2		

Figure 4: Editing the Samples.csv file in spreadsheet program Microsoft Excel

2. A file referred to as the targetsTable (often called *Targets.csv*; the file must have the filename extension ".csv" but the name of the file is specified in the properties file, described below) contains the list of subgroups, and details the specified targets to be matched in the synthesized list of units produced by the program.

The first line contains a set of column headers indicating the attribute labels for the targets to be matched in each zone. Additional suffixes are used to indicate special kinds of targets: the '-A' suffix is used to indicate an average value rather than a total; and the '-D' suffix is used to indicate a value where double precision is to be used by the program (rather than integer precision).

The first (left-most) column is reserved for the numerical labels for the population subgroups.

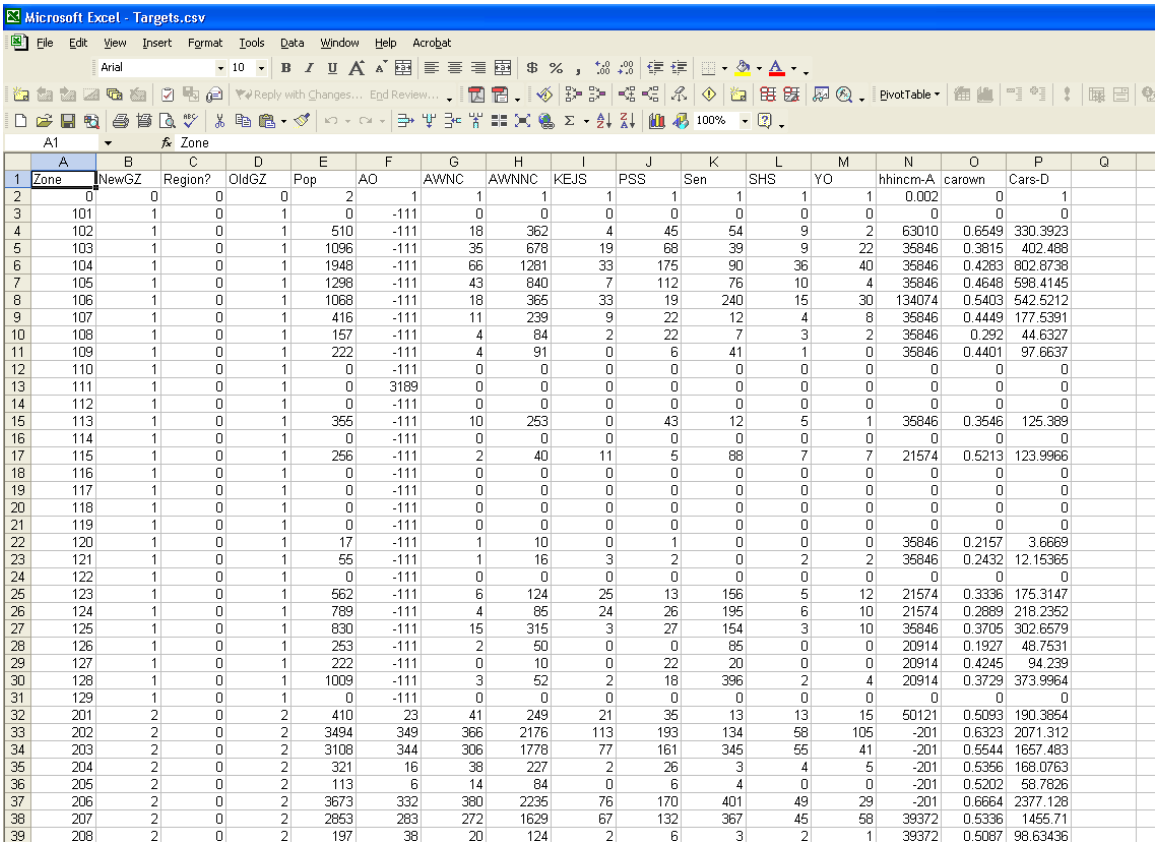
The second row contains weights for use in the calculation of the goodness-of-fit score 'gof', these are the values to use for the '$weight_a$' for the set of attributes indexed a. Note that a weight of zero completely removes the target from consideration.

Each subsequent line after the second indicates the specific target values to be matched by the program for a specific subgroup.

If a negative sign is included with a given number in the spot for the target value, then the number (the absolute value) is taken to indicate another subgroup and the corresponding value in the spot for this other subgroup is used as the target for the sum (or average, if the –A suffix is specified in row 1) of the values in the synthetic list for all the zones that include a negative sign number 'pointing' to this same positive zone. An example of the use of this negative sign is included in the screenshot below.

A screenshot showing the format of this file is included below (Figure 5). Note that in this example, the negative numbers '-111' in the column labeled AO indicate that for all of the Zones (subgroups) numbered between 101 and 129 (rows 3 through 31 in the file) the total quantity target for the AO attribute should be 3189 (specified in row 31 for Zone 111).

Note also that for the combined population in Zones 201, 202, 203, 204, 205 and 206 the hhincm average should be 50121, as specified in the column labeled hhincm-A in rows 32 through 37 in the file.



Microsoft Excel - Targets.csv

Zone	NewGZ	Region?	OldGZ	Pop	AO	AWNC	AWNNC	KEJS	PSS	Sen	SHS	YO	hhincm-A	carown	Cars-D
0	0	0	0	2	1	1	1	1	1	1	1	1	0.002	0	1
101	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
102	1	0	1	510	-111	18	362	4	45	54	9	2	63010	0.6549	330.3923
103	1	0	1	1096	-111	35	678	19	68	39	9	22	35846	0.3815	402.488
104	1	0	1	1948	-111	66	1281	33	175	90	36	40	35846	0.4283	802.8738
105	1	0	1	1298	-111	43	840	7	112	76	10	4	35846	0.4648	598.4145
106	1	0	1	1068	-111	18	365	33	19	240	15	30	134074	0.5403	542.5212
107	1	0	1	416	-111	11	239	9	22	12	4	8	35846	0.4449	177.5391
108	1	0	1	157	-111	4	84	2	22	7	3	2	35846	0.292	44.6327
109	1	0	1	222	-111	4	91	0	6	41	1	0	35846	0.4401	97.6637
110	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
111	1	0	1	0	3189	0	0	0	0	0	0	0	0	0	0
112	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
113	1	0	1	355	-111	10	253	0	43	12	5	1	35846	0.3546	125.389
114	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
115	1	0	1	256	-111	2	40	11	5	88	7	7	21574	0.5213	123.9966
116	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
117	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
118	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
119	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
120	1	0	1	17	-111	1	10	0	1	0	0	0	35846	0.2157	3.6669
121	1	0	1	55	-111	1	16	3	2	0	2	2	35846	0.2432	12.15365
122	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
123	1	0	1	562	-111	6	124	25	13	156	5	12	21574	0.3336	175.3147
124	1	0	1	789	-111	4	85	24	26	195	6	10	21574	0.2889	218.2352
125	1	0	1	830	-111	15	315	3	27	154	3	10	35846	0.3705	302.6579
126	1	0	1	253	-111	2	50	0	0	85	0	0	20914	0.1927	48.7531
127	1	0	1	222	-111	0	10	0	22	20	0	0	20914	0.4245	94.239
128	1	0	1	1009	-111	3	52	2	18	396	2	4	20914	0.3729	373.9964
129	1	0	1	0	-111	0	0	0	0	0	0	0	0	0	0
201	2	0	2	410	23	41	249	21	35	13	13	15	50121	0.5093	190.3854
202	2	0	2	3494	349	366	2176	113	193	134	58	105	-201	0.6323	2071.312
203	2	0	2	3108	344	306	1778	77	161	345	55	41	-201	0.5544	1657.483
204	2	0	2	321	16	38	227	2	26	3	4	5	-201	0.5356	168.0763
205	2	0	2	113	6	14	84	0	6	4	0	0	-201	0.5202	58.7826
206	2	0	2	3673	332	380	2235	76	170	401	49	29	-201	0.6664	2377.128
207	2	0	2	2853	283	272	1629	67	132	367	45	58	39372	0.5336	1455.71
208	2	0	2	197	38	20	124	2	6	3	2	1	39372	0.5087	98.63436

Figure 5: Example targets file

3. The .properties file, commonly called *Synthesizer.properties* (must have the file extension ".properties".) This file is a text file containing the specification of the required run-time inputs for the program. Each line in this file contains two elements, a property name and then a property value. The property name and its corresponding values are separated by an equals sign.

The property names are:

inputLocation = the directory containing the input files for the program.

samplesTable = the name of the file containing the samples, without the .csv extension (e.g. if samplesTable=samples then the program will look for the file called “samples.csv”).

targetsTable = the name of the file listing the subgroups and describing the targets and the weights associated with each target, without the .csv extension.

maxIterations = the number of iterations to proceed before stopping.

gofDifExponent = the parameter γ in the probability equation shown above.

coolingParameter = the parameter α in the probability equation shown above.
initialGenerationBasis = the basis for the initial population to be used at the start of the operations. If the word “previous” (without the quotes) is specified, then the resulting population from a previous run of the program – stored in the file named “Output******.csv” as described below – is used. If the name of one of the columns is specified, then the values for that column name are used to generate an initial population before any operations are tried. In this process samples are randomly added to each subgroup until the total for this column meets or exceeds the target specified for this column. Note that a target is required for each row in this column, i.e. no group targets can be specified for this column using negative signs. If this property is not specified then the second column in the targetsTable is used to generate the initial population.

iterationsPerZonalLackOfFit = ι in the equations above, controlling the number of operations attempted in each subgroup before proceeding to the next subgroup.

A screenshot showing the format of this file is included below (Figure 6).

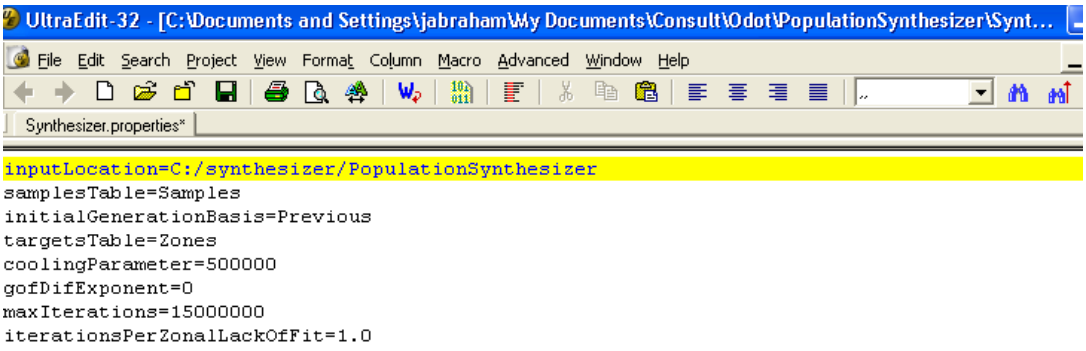


Figure 6: Editing Synthesizer parameters in UltraEdit

4. A file controlling the output of debug logging statements is required. This file is normally called log4j.xml and an example is included in the distribution. Documentation on the format of this file is available at http://logging.apache.org/ It is not normally necessary to change the content of this file.

Outputs

*Output******.csv* (with ****** in this name the same as the name specified for the samplesTable file, which is “Samples” in the screenshot example shown above). This file contains the final list of units produced by the program. The file consists of two columns, the first column is called “Zone” and identifies the subgroup of the population using the same numbers as in the leftmost column of the targetsTable. The second column is called “UnitID” and identifies a sample using the same numbers as in the leftmost column of samplesTable.

Fit.txt This file reports on the extent that the final synthetic list of units produced by the program matches the specified targets. It shows for each subgroup the specified target values and corresponding achieved values and goodness-of-fit for each target, along with the gof score for each subgroup.

Running the Program

The program is a Java program, and can be run in a number of ways. The program requires Java 1.5 (also known as Java 5), which should be downloaded and installed from java.sun.com. To check whether Java version 1.5 is installed open a windows CMD prompt and type “java –version”.

A windows command file called “synthPop.cmd” is included in the distribution. This file contains the command shown below. In Windows, if all of the input files and synthPop.cmd are all located in the same directory, the program can be run by simply double-clicking on synthPop.cmd.

Windows is not required to run the program; the program should run on any operating system with a Java 1.5 runtime environment. The program has only been tested under Windows.

The program itself is contained in the following .jar files:

Synthesizer.jar
common-base.jar
log4j-1.2.9.jar

The synthPop.cmd file contains the following command:

java -Dlog4j.configuration=log4j.xml -cp Synthesizer.jar;common-base.jar;log4j-1.2.9.jar com.hbaspecto.synthesizer.GeneratePopulation *Synthesizer.properties*

where *Synthesizer.properties* is the name of the properties file.

License